Harvest Project: La Vie
July - October, 2012
Josef Stefan Institute, Ljubljana
PASCAL & Harvest

- **PASCAL** = Pattern Analysis, Statistical Modeling and Computational Learning
  - A Network of Excellence funded by the EU
  - Promotes the use of ML in domains such as:
    - Machine Vision
    - Speech
    - Haptics
    - Brain-Computer Interface
    - Natural Language Processing
    - Information Retrieval
    - Textual Information Access
    - Multimodal integration

- **Harvest Programme**
  - Demanding channel to increase the impact of PASCAL on society and the economy
  - Applied research projects by teams of 4-8 persons for a duration of 30-90 days
  - Some piece of software as the main objective
Project La Vie

- PASCAL Harvest founded project
  - La Vie = Learning Adapted Video Information Enhancer
- Main goal:
  - To provide users with recommendations on suitable lectures for their needs
- Key components:
  1. Text extraction and information retrieval
  2. Enrichment
  3. Topic and user modeling
  4. Recommendation
  5. Visualization
- We concentrated on English language only!
Text extraction

- Sets of scripts:
  - Retrieving metadata information from internal VL database
  - Retrieving textual information from Wikipedia, DBLP and Google (abstracts and/or articles)
  - Extracting text from slides (PPT, PDF or JPGs using OCR)
  - Extracting text from transcriptions

- Each lecture is represented as:
  - BoW - Bag of words (from text extraction)
  - BoC - Bag of categories (categories that a particular lecture belongs to)

- Reduced dictionary size from approx. 2 million to 300.000 words
  - Filtering out words that appear in less than 3 different lectures
Enrichment

- Using Enrycher
  - See http://enrycher.ijs.si/
- Trained with our data and taxonomy (categories)
- Proved to be not very usable
  - Categories specified manually by VL admins are much better than automatic categorization
  - Not many usable entities or keywords returned
- Using Enrycher would only make sense if manual tagging was not possible
Topic and user modeling (1)

- User’s history
  - Set of lectures a user has seen
    (represented by a BoW and BoC computed over all lectures that user has seen)

- Lecture content
  - Semantically similar lectures

- Collaborative filter
  - Users that viewed similar lectures
Topic and user modeling (2)

7 features:

1. Lecture popularity
   - Number of visits

2. Content similarity
   - \( \text{BoW}(L_c) \cdot \text{BoW}(L_p) \)

3. Category similarity
   - \( \text{BoC}(L_c) \cdot \text{BoC}(L_p) \)

4. User content similarity (computed on the fly)
   - \( \text{BoW}(\text{Hist}(U)) \cdot \text{BoW}(L_p) \)

5. User category similarity (computed on the fly)
   - \( \text{BoC}(\text{Hist}(U)) \cdot \text{BoC}(L_p) \)

6. Co-visits
   - Number of times of \( L_c \) and \( L_p \) viewed in the same browsing session

7. User similarity
   - Number of users who have watched both \( L_c \) and \( L_p \)
Speedups

- Most of the data from the database is stored in Web service memory
  - Currently around 9 GB
- Lecture similarity features (2, 3, 6 and 7, from the biggest table) are being retrieved from the database
  - Speedup using the PostgreSQL CLUSTER command (query with approx. 10,000 rows: 27 s → 6 ms)
- Distributing load between 2 or more instances of the Web service
  - Using **Pound** load-balancer
Recommendation (1)

- Using SVM classifier for training:
  - Positive samples: two months of clicks using current recommender
  - Resulting feature weights:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecture popularity</td>
<td>-0.00003</td>
</tr>
<tr>
<td>Content similarity</td>
<td>0.00452</td>
</tr>
<tr>
<td>Category similarity</td>
<td>0.00148</td>
</tr>
<tr>
<td>User content similarity</td>
<td>0.02724</td>
</tr>
<tr>
<td>User category similarity</td>
<td>0.04167</td>
</tr>
<tr>
<td>Co-visits</td>
<td>0.00187</td>
</tr>
<tr>
<td>User similarity</td>
<td>0.01519</td>
</tr>
</tbody>
</table>
Final recommendation

- A linear SVM classifier was used to rank all possible recommendation links:

Given $L_c$ and $U$:

For all $L_p \neq L_c$:

$$\tilde{x} \quad ... \quad \text{feature vector computed for the triplet } (L_c, L_p, U)$$

$$\text{score}(\tilde{x}) = \bar{w} \cdot \tilde{x} = \sum_{n=1}^{7} w_n \cdot x_n$$

- Lectures with top 10 scores are recommended
Recurring tasks

- **Daily update**
  - At night, updates database with
    - New lectures added to VL
    - New users (both registered and anonymous)
    - New user history (new lectures viewed)
    - Removing anonymous users being offline for more than 14 days (expired cookies)

- **Monthly update**
  - Once per month or whenever a considerable amount of lectures have been added to VL
  - Generates a new fixed vocabulary
  - A new database is created (this task requires approx. 3 days)
Evaluation

- Evaluation
- Using coin flipping between old and new recommender
- Counting the number of clicks
- Try http://dev.videolectures.net/
- Our recommendations have /?ref=r00: in links
Visualization

- Using Document Atlas
  - Showing clusters of similar lecture categories
  - Size of dots depends on number of visits
  - Clicking on a dot opens a list of lectures from that category

Try [http://scienceatlas.ijs.si/videoatlas/](http://scienceatlas.ijs.si/videoatlas/)